Inspecting and Repairing Physical Topology in a Moving Grid Grain Growth Simulation[1]

Andrew Kuprat

Theoretical Division, Group T-1, Mailstop B-221
Los Alamos National Laboratory
Los Alamos, NM 87545
kuprat@lanl.gov

Abstract

Simulation of metallic grain growth in 3-D is performed by using a moving grid of tetrahedra and conforming interfacial triangles. Each metallic grain is discretized using several hundred tetrahedra in the Los Alamos grain growth simulation code *Grain3D* [1], with the support of the Los Alamos unstructured grid toolbox, *LaGriT* [2]. As the interfaces evolve by mean curvature motion, the grains coarsen: some grains grow at the expense of others, and the total number of grains decrease as a function of time. The disappearance of grains is one of several topological changes that must be undertaken by the simulation. This paper describes the algorithms used to perform topological changes in the grain growth simulation. It is hoped that some of these algorithms will have application in other fields beyond grain growth simulation, such as etching and deposition simulations where complex topological changes are modeled.

Introduction

Problematic physical topologies in metallic microstructure simulations can be classified as "illegal" or "disappearing". An example of an illegal topology is when 5 or more materials meet at a single point in 3D space. More precisely, due to energy minimizing considerations, such a topology does not occur in a soap froth [3]; our grain growth simulation *Grain3D*, which uses a mean curvature motion physical model, is governed by the same energy considerations and obeys the same topological restrictions. An example of a "disappearing" topology is when a grain is small and is getting smaller. In a short amount of time, the grain will disappear and the simulation topology will have to change. The case of illegal topologies is handled by

[1] This research is supported by the Department of Energy under contract W-7405-ENG-36.

the **popcones** algorithm and the case of disappearing topologies is handled by the **popcomponents** algorithm.

In **popcones**, we examine all vertices and look at all the incident material "cones". (In a small neighborhood of a particular vertex V, incident materials will have the topology of several cones meeting at a point.) If the connectivity of the cones is deemed to be illegal, we repair the topology (i.e., "pop" the cones). To limit the effects of this repair process to a small neighborhood of vertex V, new vertices are created close to V and new small tetrahedra are created which are incident on V. We will describe the algorithms used.

In **popcomponents**, we loop through the unstructured tetrahedral mesh and compute a data structure which reflects the physical topological components: regions (grains), interface surfaces, intersection lines where three or more regions touch, and special points where intersection lines meet. The data structure relates the various topological components using boundary operators which map topological components to neighboring components that differ in codimension by ± 1. **Popcomponents** then uses geometrical criteria, such as sizes and collapse rates of the various components, in order to determine which components are disappearing. The topological data structure is then consulted to determine an alternate topology which will replace the disappearing topology. As in **popcones**, the repair process involves insertion of nodes and creation of small tetrahedra in the neighborhood of the changing components.

Finally, we will show some images from recent simulations where *Grain3D* has been used to simulate a system of 650 grains, involving thousands of topological components and hundreds of thousands of tetrahedra.

Removal of Illegal Physical Topologies in the Tetrahedral Mesh

Long ago, Plateau [4], after numerous experimental observations, conjectured that only certain topologies were allowed in a soap froth. Specifically, lines of intersection of soap films were supposed to be formed when three films intersected, each making an angle of 120^o with each other. In turn, points at which "triple lines" intersect where supposed to be formed when four lines intersect with each other, each pair of lines making the tetrahedral angle (109^o) with each other. (The conjecture is different when the soap films intersect an external boundary.) Much later, Plateau's requirements were rigorously proven by Jean Taylor using the methods of geometrical measure theory [3]. In our grain growth simulation, we need to enforce the topological aspects of Plateau's requirements. The Gradient Weighted

Moving Finite Element method [5][6][7] used to continuously move the surfaces by mean curvature will ensure that the angles the surfaces and triple lines make with each other are correct *given* that the grid topology is correct. To enforce the correct grid topology, we use the **popcones** algorithm.

Action taken to establish Plateau's requirements

Each grain (i.e. material region separated from adjacent regions by surfaces) is given a unique "material" number. Although technically we are modeling grains that may all be made of the same material (i.e. aluminum) and which differ in crystal orientation, for our purposes we call each grain a different "material".

When a mesh component c (an edge e or a node i) fails Plateau's requirements, there are usually "too many" material components incident upon c; the crowding of components at c forces some components to not be adjacent to each other. The typical cases are four materials incident on an edge and five materials incident on a node. In this case, we will perform actions to *detach* one of the components from c. This is the origin of the name "popcones". The material components incident on c are in the shape of cones (with polygonal bases) if c is a node. If c is an edge, the incident material components projected on a plane orthogonal to the edge are 2-D cones (i.e. pie slices). **Popcones** fixes illegal topologies by detaching or "popping" one of the material cones.

To determine this "losing" component, we calculate the angle subtended by each component. For the case of a component incident on an edge, we add up the dihedral angles of all the tetrahedra that constitute that component to arrive at an angle for the component. In the case of a component incident on a node i, we add up the solid angles of all the constituent tetrahedra at node i. The heuristic we use is that the losing component—the component that will be detached—will be the one subtending the smallest angle. This is reasonable, since the surfaces bounding each component exert surface tension forces and the component subtending the smallest angle will have the greatest alignment of forces, leading to the greatest tendency to "pull away" from c.

The method of detaching the losing component (call it c_L) will be by "recoloring" (changing the material of) c_L in a very small neighborhood of c. The neighborhood is to be confined to be within a cylinder of radius `cut_length` with e as axis or a ball of radius `cut_length` with i as origin. The parameter `cut_length` is setable by the user and smaller values lead to smaller time errors in the motion of the surfaces in our simulation. The small amount of c_L that is "lost" will be replaced by an adjacent "winning" component c_W. Our heuristic is to choose c_W to be the component adjacent

to c_L which subtends the largest angle.

To ensure that all recoloring occurs in the required small ball or cylinder, we loop through all the edges e_L in the mesh that are incident on c and which themselves contain component c_L. Let $e_L = \overline{n_1, n_2}$, where n_1 is in c (either an endpoint of e or the node i) and n_2 is not in c. If n_2 is has distance $>$ `cut_length` from c, we refine e_L at its midpoint, or at a point a distance `cut_length` from c, whichever is closer.

If no refinements are necessary, we "recolor" c_L; we change all the material tags of the tetrahedra in c_L to the material of c_W. However, if refinements *are* necessary, we perform them and then recolor the resulting smaller tetrahedra.

We call **popcones** whenever the topology of the mesh has been changed for any reason, in order to *recertify* that the mesh topology satisfies Plateau's requirements.

Removal of "Disappearing" Physical Toplogies in the Tetrahedral Mesh

Even if a mesh formally satisfies the Plateau requirement, there may be topological features that are very small that need to be removed. Since the GWMFE code is able to move surfaces but is not able to change topologies, it will shrink volumes (grains) to nearly zero volume, surfaces (grain interfaces) to nearly zero area, and intersection lines (grain triple junctions) to nearly zero length. If the tiny topological components are not removed or "popped", the computational grid will effectively be stagnated at a topology that does not mirror the physical topology in real life. The command that removes the disappearing topological components is called **popcomponents**.

Popcomponents data structure

The tetrahedral grid contains volumes, surfaces, and intersection lines defined by the material tags of the tetrahedra and by the external surfaces. I.e., the triangles between adjacent tetrahedra of differing materials are parts of surfaces, and so are those triangular faces that appear on the external surfaces. In order to be able to do its work, **popcomponents** must compile a list of all the topological components (volumes, surfaces, intersection lines, and special points that are at the intersection of intersection lines), and it must somehow store the relationships between these components. This is done by computing the topological components in ascending order of codimension.

We start by finding the topological components that have codimension zero—the volumes. We start with the first tetrahedron in the mesh T_1, we note its material m, and then we use the tetrahedral element-element relation to find all of its face neighbors which also are of material m. We proceed recursively until we have found the entire component consisting of tetrahedra that have material m and are connected to T_1 through the element-element relation. (This is the maximal uniform-material face connected set of tetrahedra containing T_1.) We call this component c_1. We then find the first tetrahedron that is not in c_1 and then derive a new component c_2 that contains that tetrahedron. We continue in this fashion until we find the last component c_{n_v}. At this point we have found the n_v grains or volumes that comprise the computational mesh. These are the first n_v topological components—all the components of codimension zero.

During the compilation of the components of codimension zero, we keep track of the triangles found that separate these components from each other and from the external surfaces. We now take the first such triangle Δ_1 and, for each of its three edges, find if there are unique triangles that share those edges with Δ_1. We proceed recursively until we have found the entire component consisting of triangles that are connected to Δ_1 through "bivalent-edge-connected paths". (Precisely, we say that Δ_a and Δ_b are bivalent-edge-connected if there is a path $\Delta_a \equiv \Delta^{(1)}, \ldots, \Delta^{(m)} \equiv \Delta_b$ such that each consecutive pair of triangles $\Delta^{(i)}$, $\Delta^{(i+1)}$ share an edge and they are the only two triangles that share that edge.) We call this component c_{n_v+1}. Starting with the next unclaimed triangle, we generate component c_{n_v+2}, and proceed in this fashion until we have used up the last triangles with component $c_{n_v+n_s}$. At this point we have found the n_s surfaces contained in the computational mesh. These are all the components of codimension one. (Note: for purposes of this algorithm, edges separating distinct exterior surfaces, such as the top and side of a box, are not counted as being bivalent. This ensures that the exterior surfaces will each be separate components.)

During the compilation of the components of codimension one, we keep track of the non-bivalent edges, and generate components $c_{n_v+n_s+1}, \ldots,$ $c_{n_v+n_s+n_l}$. Each of these components consist of non-bivalent edges that can be strung together end-to-end, and which terminate at "non-bivalent nodes": nodes which have less or more than two incident non-bivalent edges. At this point we have found the n_l intersection lines contained in the computational mesh. These are the components of codimension two.

Finally, the n_p nodes which terminate the n_l intersection lines are called components $c_{n_v+n_s+n_l+1}, \ldots, c_{n_v+n_s+n_l+n_p}$. These are the n_p "special points" contained in the computational mesh which are the components of codimension three.

(We note there that it is clear that this algorithm could be generalized to finding the topological components of a face connected collection of simplices (possessing material tags which implicitly define the components) in dimensions greater than three.)

Now while we are assembling a particular component c, we record which lower codimensional (i.e. higher dimensional) components are bounded by c. That is, we record which volumes are on either side of a surface component, which surfaces are incident on an intersection line component, or which intersection lines are incident on a special point component. We can thus define the boundary operator ∂ and the inverse boundary operator ∂^{-1}. For a component c, ∂c is the set of components that are incident on c and which have dimension one lower than that of c. $\partial^{-1}c$ is the set of components incident on c which have dimension one higher than that of c. So for example, if c were a box, then ∂c would be the six faces of the box, and if $d \in \partial c$ were one of the faces, ∂d would be the set of the four bounding edges of the face. Further, if A is a set of components, we define

$$\partial A \equiv \cup_{c \in A} \partial c$$

and

$$\partial^{-1} A \equiv \cup_{c \in A} \partial^{-1} c.$$

Further we define

$$\partial^2 c \equiv \partial \circ \partial$$

and similarly we define ∂^3,∂^{-2}, and ∂^{-3}. So again for the example of c being a box, then $\partial^2 c$ would be the 12 edges of the box. If b were a corner, $\partial^{-1} b$ would be the three edges incident on b and $\partial^{-2} b$ would be all the faces incident on all the edges incident on b, which would be the three faces incident on b.

Kinematical and geometrical criteria for removal of components

Popcomponents needs to evaluate whether a component c is "disappearing", and if so, it must take action to remove it.

Our primary criteria is kinematical. The GWMFE method computes a velocity field $\dot{\mathbf{x}}_i$ at each node i. Thus we can define the rate of growth or shrinking of the length, surface area, or volume of c. Since we know the length, surface area, or volume of c, by simple linear extrapolation, we can compute an estimated collapse time if c is shrinking. If the estimated collapse time is within a user defined tolerance `toldt` of the present, we will take action to remove c.

This was the only component removal criterion used in [1]. Although the node velocities for the most part are defined by the physical system—curvature forces and uniform isotropic viscous resistance to motion—there are non-physical "regularization" forces as well. When a component is collapsing and the topology does not change, the regularization terms kick in when c has collapsed down to a very small diameter. Without these terms, the tetrahedra comprising or bounding c would eventually have zero or negative volume, and the simulation would terminate. This means that the primary criteria can sometimes fail to indicate that removal is necessary—the collapse rate of c has declined to nearly zero when c has a tiny diameter, and hence the estimated collapse time is long or infinite. We have thus devised a secondary (geometrical) collapse criterion to signal removal of these "stagnated" topologies.

First we compute an estimate of the diameter of c, and if this number is greater than a user defined tolerance `tol_small`, we conclude that the geometrical criterion also indicates that no removal is necessary. If the diameter is $<$`tol_small` it may be possible that c was just freshly created and that it would remain and/or expand under the action of the actual PDE that we are trying to solve. If c is a volume, this is unlikely since it is known that a grain would have to have at least about 14 faces in order to be expanding, but grains with tiny diameter $<$ `tol_small` are never observed to have that many faces in our simulation. Thus, if c is a volume with diameter $<$ `tol_small` we say the geometrical criterion is satisfied and we flag c for removal.

If c is a surface or an intersection line of small size, there is a reasonable chance that c is indeed stable or expanding. Thus the geometrical criterion for removal of these components also includes a heuristic involving the solid angles of the volume components incident on c. To understand this heuristic, we have to divide the volume components incident on c into two sets: c_+ and c_-.

Suppose c is an m-sided polygonal surface which may be expanding or collapsing. The m sides of c are none other than ∂c. c bounds two volumes—these volume regions are $\partial^{-1}c$. Finally, there are m volumes around the perimeter of c which we call the peripheral volumes. This set of volumes are in fact the set $(\partial^{-2}\partial \setminus \partial^{-1})c \equiv \partial^{-2}\partial c \setminus \partial^{-1}c$. That is, the peripheral volumes are all the volumes incident on the boundary of c minus the set of volumes that c itself bounds. We refer to the bounded set of volumes as c_+ and the peripheral set as c_-. That is

$$\begin{aligned} c_+ &\equiv \partial^{-1}c \\ c_- &\equiv (\partial^{-2}\partial \setminus \partial^{-1})c. \end{aligned}$$

Now suppose c is an intersection line which bounds m surfaces $(\partial^{-1}c)$,

which themselves bound the same number of volumes ($\partial^{-2}c$). We say these volumes are *directly* incident on c. The peripheral set of volumes (at either end of c) are those volumes which are incident on ∂c but are not directly incident on c. Denoting the directly incident set of volumes by c_+ and the peripheral set by c_-, we have

$$\begin{aligned} c_+ &\equiv \partial^{-2}c \\ c_- &\equiv (\partial^{-3}\partial \setminus \partial^{-2})c. \end{aligned}$$

Now we give the heuristic for removal of surface and line components c that have diameter $<$ `tol_small` but which are not shrinking. Since c has very small diameter, it appears to virtually be a point. So the components in $c_+ \cup c_-$ will appear to coincide at a point whose material component connectivity probably does not satisfy the Plateau requirement. In this case, the assumption of **popcones** was that the component c_L subtending the smallest solid angle at the point c would be the one to "pull away" and would be the one disconnected from c. Since c is "nearly" a point, we can in fact measure the solid angles subtended by the volume components in c_+ and c_-. If the smallest solid angle is subtended by a directly incident volume c_L in c_+, we conclude the pulling away of c_L would cause c in fact to shrink and hence that the component c should be flagged for removal. If however we find that the component c_L subtending the smallest solid angle is in the set of peripheral volumes c_-, we conclude that the pulling away of c_L would cause c to in fact grow, and hence that c should *not* be flagged for removal.

Action taken to remove components flagged for removal

Suppose c is a surface or intersection line that is flagged for removal. In the **popcones** algorithm, we allowed the component c_L to pull away from the point i of illegal topology by recoloring a small portion c_L at i. In fact we replaced the material tags of the tetrahedra of c_L incident on i by the material tags of the "winning component" c_W which was the component adjacent to c_L that subtended the largest solid angle. If some of the tetrahedra in c_L that were incident on i had edges at i that were longer than the user-defined length `cut_length`, we refined the tetrahedra so that only tetrahedra within `cut_length` of i had to be recolored. Analogous to that we chose the losing component c_L to be the component in c_+ subtending the smallest solid angle. We choose the winning component c_W to be component in c_- that subtends the largest solid angle. Having chosen a winning component c_W, as with **popcones**, we recolor the tetrahedra in c_L incident on c with the material type of c_W, but only after we have performed any

necessary refinements to ensure that all edges of these tetrahedra that have one endpoint in c and one endpoint not in c have length $<$ `cut_length`. This recoloring of c_L by c_W allows c_L to pull away and effectively replaces c with a new and hopefully expanding topology. It is easy to verify with a few diagrams that the annihilation of a three-sided surface with this recoloring leads to the creation of a triple line at the intersection of the three formerly peripheral components, and the annihilation of a triple line with one peripheral volume component at each end with this scheme results in the creation of a three-sided surface separating the two formerly peripheral volumes. These two actions which are inverses of each other are very commonly observed in soap films [8].

In the case that c is a collapsing volume, we recolor that small tetrahedra that comprise c with the material color of one of components in $c_- \equiv \partial^{-1}\partial c \setminus c$, the set of volumes incident on ∂c with c removed. Again, since the diameter of c is small, we can effectively treat it like a point and evaluate the solid angles subtended by the components of c_-. We use the heuristic of choosing the material tag of the component in c_- subtending the largest angle at c for the recoloring of c.

Sample Numerical Runs

We have been running our microstructure simulations on long rectangular domains which represent simulations of grain growth in the tiny metal (usually aluminum) wires present in semiconductor devices. The simulations can involve thousands of grains, each comprised of hundreds of tetrahedra. Since this implies there are tens of thousands of surfaces between all the various grains, it is true that topological change is constantly occurring. In Figure 1 we see a portion of a $1 \times 1 \times 50$ line that started with 650 grains and has already evolved down to having just a few hundred grains at $t = 0.1$. (Eventually, at long times, the desirable asymptotic microstructure is "bamboo" where all grain interfaces run perpendicular to the direction of the wire. This kind of microstructure is less susceptible to component failure.) In Figure 2 we that the portion of the line has undergone visible topological change. The external top and front faces show that grains have disappeared or changed neighbors. Of course, this kind of (two-dimensional) illustration fails to show the complex topological changes that have been modeled in the *interior* of the domain.

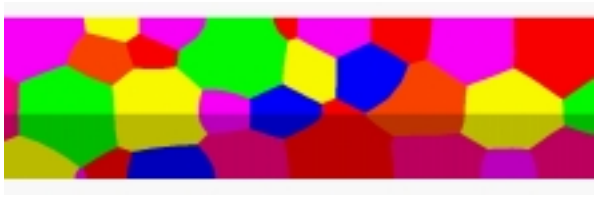

Figure 1: **Portion of** $1 \times 1 \times 50$ **line at** $t = 0.1$**.**

Figure 2: **Line at** $t = .15$ **with topological change visible on top and front surfaces.**

References

[1] A. Kuprat, *Modeling Microstructure Evolution using Gradient-Weighted Moving Finite Elements,* SIAM J. Sci. Comput., in press.

[2] D.C. George, *LaGriT User's Manual,* http://www.t12.lanl.gov/~lagrit.

[3] Jean E. Taylor, *The structure of singularities in soap-bubble-like and soap-film-like minimal surfaces,* Annals of Math., Vol. 103, pp. 489-539, 1976.

[4] J.A.F. Plateau, *Statique Experimentale et Theorique des Liquides Soumis aux Seules Forces Moleculaires,* Gauthier-Villiard, Paris, 1873.

[5] M.J. Baines, *Moving Finite Elements,* Oxford University Press, 1994.

[6] N. Carlson and K. Miller, *Design and Application of a Gradient-Weighted Moving Finite Element Code I: in One Dimension,* SIAM J. Sci. Comput., Vol. 19, pp. 728-765, 1998.

[7] N. Carlson and K. Miller, *Design and Application of a Gradient-Weighted Moving Finite Element Code II: in Two Dimensions,* SIAM J. Sci. Comput., Vol. 19, pp. 766-798, 1998.

[8] A.C. Ferro and M.A. Fortes, *The Elimination of Grains and Grain Boundaries in Grain Growth,* Interface Sci., Vol. 5, pp. 263-278, 1997.